In the Specification:

Please amend the title as follows:

COMPUTER~~ING~~ MACHINE HAVING IMPROVED ~~COMPUTING ARCHITECTURE~~ DATA TRANSFER AND RELATED SYSTEM AND METHOD

Please amend the specification as follows:

[2] This application is related to U.S. Patent App. Serial Nos. 10/684,102 entitled IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-11-3)~~, 10/683,929 entitled PIPELINE ACCELERATOR FOR IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-13-3)~~, 10/684,057 entitled PROGRAMMABLE CIRCUIT AND RELATED COMPUTING MACHINE AND METHOD ~~(Attorney Docket No. 1934-14-3)~~, and 10/683,932 entitled PIPELINE ACCELERATOR HAVING MULTIPLE PIPELINE UNITS AND RELATED COMPUTING MACHINE AND METHOD ~~(Attorney Docket No. 1934-15-3)~~, which have a common filing date and owner and which are incorporated by reference.

[49] **FIG. 3** is a schematic block diagram of a computing machine ***40***, which has a peer-vector architecture according to an embodiment of the invention. In addition to a host processor ***42***, the peer-vector machine ***40*** includes a pipeline accelerator ***44***, which performs at least a portion of the data processing, and which thus effectively replaces the bank of coprocessors ***14*** in the computing machine ***10*** of **FIG. 1**. Therefore, the host-processor ***42*** and the accelerator ***44*** are "peers" that can transfer data vectors back and forth. Because the accelerator ***44*** does not execute program instructions, it typically performs mathematically intensive operations on data significantly faster than a bank of coprocessors can for a given clock frequency. Consequently, by combing the decision-making ability of the processor ***42*** and the number-crunching ability of the accelerator ***44***, the machine ***40*** has the same abilities

as, but can often process data faster than, a conventional computing machine such as the machine ***10***. Furthermore, as discussed below and in previously cited U.S. Patent App. Serial No. ——10/683,929 entitled PIPELINE ACCELERATOR FOR IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD~~ (Attorney Docket No. 1934-13-3)~~, providing the accelerator ***44*** with the same communication interface as the host processor ***42*** facilitates the design and modification of the machine ***40***, particularly where the communications interface is an industry standard. And where the accelerator ***44*** includes multiple components (*e.g.*, PLICs), providing these components with this same communication interface facilitates the design and modification of the accelerator, particularly where the communication interface is an industry standard. Moreover, the machine ***40*** may also provide other advantages as described below and in the previously cited patent applications.

[50] Still referring to **FIG. 3**, in addition to the host processor ***42*** and the pipeline accelerator ***44***, the peer-vector computing machine ***40*** includes a processor memory ***46***, an interface memory ***48***, a bus ***50***, a firmware memory ***52***, optional raw-data input port[[s]] ***54***~~ and ***56***~~, processed-data output port[[s]] ***58***~~ and ***60***~~, and an optional router ***61***.

[52] The pipeline accelerator ***44*** is disposed on at least one PLIC (not shown) and includes hardwired pipelines $74_1 - 74_n$, which process respective data without executing program instructions. The firmware memory ***52*** stores the configuration firmware for the accelerator ***44***. If the accelerator ***44*** is disposed on multiple PLICs, these PLICs and their respective firmware memories may be disposed on multiple circuit boards, *i.e.*, daughter cards (not shown). The accelerator ***44*** and daughter cards are discussed further in previously cited U.S. Patent App. Serial Nos. ——10/683,929 entitled PIPELINE ACCELERATOR FOR IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-13-3)~~ and ———10/683,932 entitled PIPELINE ACCELERATOR HAVING MULTIPLE PIPELINE UNITS AND RELATED COMPUTING MACHINE AND METHOD~~ (Attorney Docket No. 1934-15-3)~~. Alternatively, the accelerator ***44*** may be disposed on at least one ASIC,

and thus may have internal interconnections that are unconfigurable. In this alternative, the machine ***40*** may omit the firmware memory ***52***. Furthermore, although the accelerator ***44*** is shown including multiple pipelines ***74***, it may include only a single pipeline. In addition, although not shown, the accelerator ***44*** may include one or more processors such as a digital-signal processor (DSP).

[53] The general operation of the peer-vector machine ***40*** is discussed in previously cited U.S. Patent App. Serial No. ~~__~~10/684,102 entitled IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-11-3)~~, and the functional topology and operation of the host processor ***42*** is discussed below in conjunction with **FIGS. 4 – 7**. **FIG. 4** is a functional block diagram of the host processor ***42*** and the pipeline bus ***50*** of **FIG. 3** according to an embodiment of the invention. Generally, the processing unit ***62*** executes one or more software applications, and the message handler ***64*** executes one or more software objects that transfer data between the software application(s) and the pipeline accelerator ***44*** (**FIG. 3**). Splitting the data-processing, data-transferring, and other functions among different applications and objects allows for easier design and modification of the host-processor software. Furthermore, although in the following description a software application is described as performing a particular operation, it is understood that in actual operation, the processing unit ***62*** or message handler ***64*** executes the software application and performs this operation under the control of the application. Likewise, although in the following description a software object is described as performing a particular operation, it is understood that in actual operation, the processing unit ***62*** or message handler ***64*** executes the software object and performs this operation under the control of the object.

[54] Still referring to **FIG. 4**, the processing unit ***62*** executes a data-processing application ***80***, an accelerator exception manager application (hereinafter the exception manager) ***82***, and an accelerator configuration manager application (hereinafter the configuration manager) ***84***, which are collectively referred to as the processing-unit

applications. The data-processing application processes data in cooperation with the pipeline accelerator ***44*** (**FIG. 3**). For example, the data-processing application ***80*** may receive raw sonar data via the port ***54*** (**FIG. 3**), parse the data, and send the parsed data to the accelerator ***44***, and the accelerator may perform an FFT on the parsed data and return the processed data to the data-processing application for further processing. The exception manager ***82*** handles exception messages from the accelerator ***44***, and the configuration manager ***84*** loads the accelerator's configuration firmware into the memory ***52*** during initialization of the peer-vector machine ***40*** (**FIG. 3**). The configuration manager ***84*** may also reconfigure the accelerator ***44*** after initialization in response to, *e.g.,* a malfunction of the accelerator. As discussed further below in conjunction with **FIGS. 6 – 7**, the processing-unit applications may communicate with each other directly as indicated by the dashed lines ***85***, ***87***, and ***89***, or may communicate with each other via the data-transfer objects ***86***. The message handler ***64*** executes the data-transfer objects ***86***, a communication object ***88***, and input and output read objects ***90*** and ***92***, and may execute input and output queue objects ***94*** and ***96***. The data-transfer objects ***86*** transfer data between the communication object ***88*** and the processing-unit applications, and may use the interface memory ***48*** as a data buffer to allow the processing-unit applications and the accelerator ***44*** to operate independently. For example, the memory ***48*** allows the accelerator ***44***, which is often faster than the data-processing application ***80***, to operate without "waiting" for the data-processing application. The communication object ***88*** transfers data between the data objects ***86*** and the pipeline bus ***50***. The input and output read objects ***90*** and ***92*** control the data-transfer objects ***86*** as they transfer data between the communication object ***88*** and the processing-unit applications. And, when executed, the input and output queue objects ***94*** and ***96*** cause the input and output read objects ***90*** and ***92*** to synchronize this transfer of data according to a desired priority.

[83] Still referring to **FIG. 5**, additional data-transfer techniques are contemplated. For example a single thread may publish data to multiple locations within the pipeline accelerator ***44*** (**FIG. 3**) via respective multiple channels. Alternatively, as discussed in previously cited U.S. Patent App. Serial Nos. —10/684,102 entitled

IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-11-3)~~ and ——10/683,929 entitled PIPELINE ACCELERATOR FOR IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-13-3)~~, the accelerator ***44*** may receive data via a single channel ***104*** and provide it to multiple locations within the accelerator. Furthermore, multiple threads (*e.g.*, threads $\mathbf{100}_1$ and $\mathbf{100}_2$) may subscribe to data from the same channel (*e.g.*, channel $\mathbf{104}_2$). In addition, multiple threads (*e.g.*, threads $\mathbf{100}_2$ and $\mathbf{100}_3$) may publish data to the same location within the accelerator ***44*** via the same channel (*e.g.*, channel $\mathbf{104}_3$), although the threads may publish data to the same accelerator location via respective channels ***104***.

[85] The exception manager ***82*** receives and logs exceptions that may occur during the initialization or operation of the pipeline accelerator ***44*** (**FIG. 3**). Generally, an exception is a designer-defined event where the accelerator ***44*** acts in an undesired manner. For example, a buffer (not shown) that overflows may be an exception, and thus cause the accelerator ***44*** to generate an exception message and send it to the exception manager ***82***. Generation of an exception message is discussed in previously cited U.S. Patent App. Serial No. -10/683,929 entitled PIPELINE ACCELERATOR FOR IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-13-3)~~.

[86] The exception manager ***82*** may also handle exceptions that occur during the initialization or operation of the pipeline accelerator ***44*** (**FIG. 3**). For example, if the accelerator ***44*** includes a buffer (not shown) that overflows, then the exception manager ***82*** may cause the accelerator to increase the size of the buffer to prevent future overflow. Or, if a section of the accelerator ***44*** malfunctions, the exception manager ***82*** may cause another section of the accelerator or the data-processing application ***80*** to perform the operation that the malfunctioning section was intended to perform. Such exception handling is further discussed below and in previously cited U.S. Patent App. Serial No. ———10/683,929 entitled PIPELINE ACCELERATOR FOR

IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-13-3)~~.

[95] When sent to the accelerator ***44***, the exception-handling instruction may change the soft configuration or the functioning of the accelerator. For example, as discussed above, if the exception is a buffer overflow, the instruction may change the accelerator's soft configuration (*i.e.,* by changing the contents of a soft configuration register) to increase the size of the buffer. Or, if a section of the accelerator ***44*** that performs a particular operation is malfunctioning, the instruction may change the accelerator's functioning by causing the accelerator to take the disabled section "off line." In this latter case, the exception manager ***82*** may, via additional instructions, cause another section of the accelerator ***44***, or the data-processing application ***80***, to "take over" the operation from the disabled accelerator section as discussed below. Altering the soft configuration of the accelerator ***44*** is further discussed in previously cited U.S. Patent App. Serial No. -10/683,929 entitled PIPELINE ACCELERATOR FOR IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD ~~(Attorney Docket No. 1934-13-3)~~.